AMENDMENTS TO THE CLAIMS

Kindly add new claims 75-78 as shown in the following listing of claims. The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1-63. (canceled)

64. (new) A microprocessor, comprising:

a branch target address cache (BTAC), configured to cache, for each of a plurality of previously executed branch instructions: a prediction of whether said branch instruction will be taken and is present in a cache line of instruction bytes provided by an instruction cache in response to a fetch address, a target address of said branch instruction, and a location of an opcode byte of said branch instruction within said cache line;

wherein said BTAC is further configured to provide said prediction, said target address, and said location in response to said fetch address;

an instruction buffer, coupled to receive said cache line from said instruction cache and to mark a byte in said cache line within said instruction buffer indicated by said location provided by said BTAC if the microprocessor branches to said target address provided by said BTAC based on said prediction;

an instruction decoder, coupled to said instruction buffer, configured to format said instruction bytes in said cache line into formatted instructions; and

prediction check logic, coupled to said instruction decoder, configured to indicate the microprocessor erroneously branched to said target address if said instruction decoder indicates said marked byte is in a non-opcode location within one of said formatted instructions.

65. (new) The microprocessor of claim 64, wherein said marked byte is in a non-opcode location within one of said formatted instructions because said BTAC is accessed by a virtual address version of said fetch address, whereas said instruction cache is accessed by a physical address version of said fetch address.

66. (new) The microprocessor of claim 64, wherein said marked byte is in a non-opcode location within one of said formatted instructions because said cache line was the object of self-modifying code prior to said instruction cache providing said cache line of instruction bytes in response to said fetch address.

67. (new) The microprocessor of claim 64, wherein said target address is provided by a speculative call/return stack in the microprocessor rather than said BTAC in response to an indication cached in said BTAC that said cache line includes a return instruction.

68. (new) The microprocessor of claim 64, wherein said location comprises a location of a first byte of said branch instruction within said cache line.

69. (new) A method for correcting a branch instruction misprediction in a microprocessor, the method comprising:

caching in a branch target address cache (BTAC), for each of a plurality of previously executed branch instructions: a prediction of whether said branch instruction will be taken and is present in a cache line of instruction bytes provided by an instruction cache in response to a fetch address, a target address of the branch instruction, and a location of an opcode byte of the branch instruction within said cache line;

providing said cache line to an instruction buffer, by said instruction cache in response to said fetch address;

providing said prediction, said target address, and said location by said BTAC in response to said fetch address;

branching to said target address provided by said BTAC based on said prediction;

marking a byte in said cache line within said instruction buffer indicated by said location provided by said BTAC;

formatting, by an instruction decoder, said instruction bytes in said cache line into formatted instructions; and

indicating the microprocessor erroneously branched to said target address, if said instruction decoder indicates said marked byte is in a non-opcode location within one of said formatted instructions.

70. (new) The method of claim 69, further comprising:

invalidating an entry in said BTAC caching said prediction, said target address, and said location, if said instruction decoder indicates said marked byte is in a non-opcode location within one of said formatted instructions.

71. (new) The method of claim 69, further comprising:

flushing said instruction buffer, if said instruction decoder indicates said marked byte is in a non-opcode location within one of said formatted instructions.

72. (new) The method of claim 69, further comprising:

branching the microprocessor to a current instruction pointer, if said instruction decoder indicates said marked byte is in a non-opcode location within one of said formatted instructions.

73. (new) The method of claim 69, further comprising:

flushing pipeline stages of the microprocessor above said instruction decoder, if said instruction decoder indicates said marked byte is in a non-opcode location within one of said formatted instructions.

74. (new) The method of claim 69, wherein said providing said cache line and said providing said prediction, said target address, and said location are performed substantially in parallel.

75. (new) The microprocessor of claim 64, wherein said instruction decoder is further configured to indicate whether each of said formatted instructions is a non-branch instruction, wherein said prediction check logic is further configured to indicate the microprocessor erroneously branched to said target address, if said instruction decoder indicates one of said formatted instructions is a non-branch instruction and if said instruction buffer has marked a byte of said one of said formatted instructions.

76. (new) The microprocessor of claim 64, wherein said BTAC is further configured to cache a length of each of said previously executed branch instructions, wherein said instruction decoder is further configured to determine a length of each of said formatted instructions, wherein said prediction check logic is further configured to indicate the microprocessor erroneously branched to said target address, if said length of one of said formatted instructions determined by said instruction decoder does not match said length of said one of said formatted instructions cached by said BTAC.

77. (new) The method of claim 69, further comprising:

indicating, by the instruction decoder, whether each of said formatted instructions is a non-branch instruction, after said formatting; and

indicating the microprocessor erroneously branched to said target address, if said instruction decoder indicates one of said formatted instructions is a non-branch instruction and if said instruction buffer has marked a byte of said one of said formatted instructions.

78. (new) The method of claim 69, further comprising:

caching in said BTAC, for each of each of said previously executed branch instructions a length thereof;

determining, by said instruction decoder, a length of each of said formatted instructions, after said formatting; and

indicating the microprocessor erroneously branched to said target address, if said length of one of said formatted instructions determined by said instruction decoder does not match said length of said one of said formatted instructions cached by said BTAC.